Volume 6, Number 8 • August 2000
www.elementkjournals.com/sun
US $11.00

Inside Solaris™

Tips & Techniques for users of Sun Solaris

PostgreSQL—a free SQL database for Solaris

by Paul A. Watters

Many data processing applications require some form of persistent data storage, which is usually made available in the form of a relational database management system (RDBMS). Most modern RDBMSs support the ANSI SQL-92 standard for a Structured Query Language, so that queries can be constructed and a database server interrogated for the matching result set. Access to a database using SQL is usually permitted through a dedicated client terminal (such as Oracle's sqlplus utility or Ingres's isql utility). Alternatively, many applications in high-level programming languages, such as C, Perl, and Java, can contain embedded SQL statements through which queries are passed using dedicated client libraries. For example, Perl has a standard database interface known as DBI, and Java has both one- and two-tier client class libraries known as the Java Database Connectivity Classes (JDBC).

Commercial RDBMSs are very expensive, meaning that developers and nonprofit organizations often find themselves marginalized in the race for profit. In addition, developers and students who wish to understand exactly how RDBMSs work behind the scenes are unable to call up a commercial vendor and ask to see the source code.

Fortunately, there's one enterprise-level RDBMS that shares a codebase with a commercial database system and is freely available, including source code. PostgreSQL was originally created as part of a project at the University of California at Berkeley, led by Michael Stonebreaker, who's often referred to as the "father" of relational database systems.

In this article, we will examine how to compile and install PostgreSQL for Solaris,

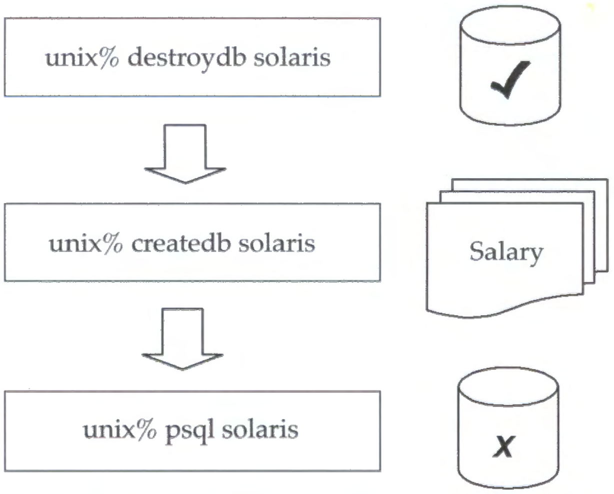

Figure A: *Creating, querying, and destroying databases is easy with PostgreSQL.*

element k JOURNALS
A Division of Element K Press

and how to create database instances and tables and execute queries. One major benefit for Solaris-based organizations is that PostgreSQL compiles and runs successfully on both Sparc and Intel platforms, making it easy to share data and standardize data processing operations on a single RDBMS. The steps involved in running PostgreSQL are summarized in **Figure A** on the cover.

History

PostgreSQL is based on Postgres 4.2, which was the last official release of the original University of California Postgres project. The Postgres project began in 1986, and has since been used as the basis for a commercial product (Computer Associates Ingres platform), as well as open source projects, such as Postgres 95 (developed by Andrew Yu and Jolly Chen). Indeed, anyone who's familiar with the commands and capabilities of Ingres will have little difficulty in cross-grading to PostgreSQL.

Postgres 95 made Postgres more accessible and acceptable by replacing the antiquated query interpreter (Postquel) with an ANSI-compliant SQL interpreter. SQL, as a query language standard, is based on the original SEQUEL query language developed by Donald Chamberlin at IBM Research. In 1996, the open source Postgres project was renamed PostgreSQL to emphasize the past connection with the original Postgres project, and the new focus on providing a standards-compliant, high-performance RDBMS.

The current production version of Postgres is 6.53. However, there's already a beta version of Postgres 7.0 available from the PostgreSQL site at **www.postgresql.org/**.

Key features

PostgreSQL supports a relational data model similar to other RDBMSs; data sets consist of tables, in which data is organized in columns and rows. Most RDBMS systems support common variable types, including character and string types (varchar), and floating point number (float). PostgreSQL extends these possibilities by including some object-oriented features, such as classes and inheritance, as well as procedural functions. In addition, PostgreSQL can act as a highly efficient transactional database by implementing some common transactional integrity checks and constraints.

PostgreSQL is also Y2K compliant, as the starting date for the database is assumed to be 1970. Thus, if dates are specified in a two-digit type, then "00" will always be interpreted as "2000", while "69" will be interpreted as "1969". Of course, this is less than useful for applications that need to process dates prior to 1970, in which case, four-digit date types are recommended.

Installing PostgreSQL

Postgres is portable to most UNIX and UNIX-type platforms that have a POSIX environment (including Windows NT), and for which the ANSI-compliant GNU development tools are available. This also means that PostgreSQL is one of the few RDBMSs that will work on both Solaris Sparc and Solaris Intel. The binary for Solaris Sparc is available from the PostgreSQL site; however, you need to compile the Solaris Intel from scratch. This is easily accomplished if you have the GNU C compiler (gcc) installed.

The Postgres super-user is the user named postgres who owns the Postgres binaries and database files. The first step in installing PostgreSQL is creating the postgres user account. All PostgreSQL applications, including the postmaster and postgres backend database engine, you must run as the postgres user. The postgres user should also belong to a group of its own (e.g., postgres), or any non-privileged, non-zero GID (such as nobody).

The postmaster is the client connection daemon that acts as the server interface for all client requests to the Postgres system. The core functions of the database server depend on the postmaster, as it manages a shared buffer pool and lock tables for multiple database instances running on the same server. For example, you could execute a database instance called accounts on the same physical system as a database called personnel, without any conflicts or security issues, as all client connections are commonly managed by the postmaster.

Since authentication is handled through the operating system, any Solaris user can potentially create and destroy his own databases. This means that, unlike some other RDBMSs, you don't need to consult a DBA to create a test or development database with normal user permissions. This reduces database administration overhead, and gives the DBA more time to devote to production issues.

You use the client program psql to issue all requests to the postmaster, and use the commands `createdb` and `destroydb` to create and destroy database instances, respectively. In addition, application developers can create C applications that use the libpq library connect to the postmaster. PostgreSQL also comes with JDBC drivers, including source, which have been widely used as the basis for JDBC support in other RDBMSs.

One possible issue that may arise when running the postmaster for the first time on Solaris Sparc is that you may get an IpcMemoryCreate error. This indicates that the maximum shared memory segment size, as set in /etc/system, isn't high enough for PostgreSQL (or any other database server, for that matter). To rectify the situation, insert the following line into /etc/system, and reboot:

```
set
shmsys:shminfo_shmmax=0x7fffffff
```

Configuring PostgreSQL

The first step in configuring PostgreSQL to run on your Solaris system is to add the binaries directory /usr/local/pgsql/bin to the postgres user's path and each user's path (e.g., by editing the sitewide .profile for the Bourne shell). If the server is on a remote machine, then you'll need to set the PGHOST environment variable to the name of the database server machine.

Next, create a startup file for PostgreSQL in /etc/init.d called *postgres*. When this file is symbolically linked into one of the run level script directories (e.g., /etc/rc2.d for run level 2), then Postgres will be started automatically after rebooting (e.g., /etc/rc2.d/S99postgres). A sample startup file might contain the following:

```
#!/sbin/sh
# postgres 6.53 startup file

case "$1" in
'start')
   su postgres -c "/usr/local/pgsql/bin/postmaster
   ➥-S -D /usr/local/pgsql/data"
   ;;

*)
   echo "Usage: $0 { start }"
   exit 1
   ;;
esac
exit 0
```

After starting Postgres with the command

```
bash# /etc/init.d/postgres start
```

you can start creating databases as an ordinary user. For example, the command

```
bash-2.03$ createdb solaris
```

creates a new database called "solaris". However, if the postmaster process fails to start for some reason, you'll see the following error message:

```
Connection to database 'solaris' failed.
connectDB() -- connect() failed: No such file or
➥directory
Is the postmaster running at 'localhost' and
accepting connections on Unix socket '5432'?

createdb: database creation failed on solaris.
```

If this occurs, then check the process list for postmaster (and verify that it's actually running as Postgres), and restart it if it isn't running. Alternatively, if it continually fails to start successfully, then you can use `lsof` to check that a process doesn't already own the port 5432.

Using PostgreSQL

If you manage to successfully create the "solaris" database, you can easily connect to it by using the psql client, which uses a similar command set to the Ingres isql client. When you start isql, you need to specify the database instance to which you wish to connect:

```
bash-2.03$ psql solaris

Welcome to the POSTGRESQL interactive sql monitor:

Please read the file COPYRIGHT for copyright terms
➥of POSTGRESQL

[PostgreSQL 6.5.3 on i386-pc-solaris2.7, compiled by
➥gcc 2.95.2]

      type \? for help on slash commands
      type \q to quit
      type \g or terminate with semicolon to
      ➥execute query
   You are currently connected to the
   ➥database: solaris
solaris=>
```

The first step in using a database is usually creating a table or set of tables in which your data will be recorded. For example, imagine a table (user_tbl) that contained employee details (firstname and surname) and pay rate (salary). We could use the following command to create the table in psql:

```
solaris=> create table user_tbl (surname
➥varchar(20), firstname varchar (10), salary
➥float(10));
```

If the table is successfully created, the server returns the response:

```
CREATE
```

Having created the table, you can now begin adding data by using the `insert` command. Add the first employee's details to the database by using the following command:

```
solaris=> insert into user_tbl values
➥('Watters','Paul',1000.00);
```

Again, if the row is successfully inserted, the server returns a response:

```
INSERT 416704 1
```

Now, add the second employee's details to the database by using the following command:

```
solaris=> insert into user_tbl values
➥('Suhm','Garrett',2000.00);
```

You'll see the following response if the row is successfully inserted:

```
INSERT 416705 1
```

If you now wanted to list all of the rows in the database, you could use the `select` command as follows:

```
solaris=> select * from user_tbl;

surname|firstname|salary
-------+---------+------
Watters|Paul     |  1000
Suhm   |Garrett  |  2000
(2 rows)
```

Of course, the asterisk wildcard (*) means that all columns are selected from the table user_tbl. However, if you only wanted to print a list of employees, without their salaries, you could use:

```
solaris=> select firstname,surname from user_tbl;

firstname|surname
---------+-------
Paul     |Watters
Garrett  |Suhm
(2 rows)
```

You can also change the order in which the rows are displayed by using the `order by` command:

```
solaris=> select * from user_tbl order by salary;

surname|firstname|salary
-------+---------+------
Watters|Paul     |  1000
Suhm   |Garrett  |  2000
(2 rows)
```

Now, if you wanted to extract a list of the highest paid employees, you could specify a limit (e.g., $1500), and select a list of all employees whose salary exceeds this limit using the command:

```
solaris=> select surname from user_tbl where
➥salary>1500;

surname
-------
Suhm
(1 row)
```

Finally, once you have decided that the table you created is no longer necessary, use the `drop` command to remove it:

```
solari=> drop table user_tbl;
```

Again, if the table is successfully dropped, the server returns a response:

```
DROP
```

Further reading

In addition to an excellent tutorial and user's guide, which are both available from the PostgreSQL Web site, there's a support mailing list that contains discussions of all issues relating to operating and installing PostgreSQL. To subscribe, send a message to Pgsql at general@postgreSQL.org. Commercial support is available from www.pgsql.com/.

Display all the users on your Solaris Box

Need a quick way to display all the users on your Solaris box? Use the `listusers` command:

```
$ listusers
bsmith
rdanson
nobody   Nobody
nobody4  SUNOS 4.x Nobody
```

Use the `-g` switch to show the users of a group:

```
$ listusers -g dba
bsmith
rdanson
```

Remember that root will not show up in a list even if it is a member of a group. You can also use the `-l` switch to see if a user (or list of users) has accounts on your system:

```
$ listusers -l bsmith,nobody

bsmith
nobody   Nobody
```

MetaFrame for Solaris—a thin-client alternative

by Clayton E. Crooks II

Citrix, which has a long-standing history of providing thin-client solutions for Windows NT servers, has recently released a new product. MetaFrame for Solaris, which is in beta, is the first release from Citrix for the UNIX platform. Despite direct competition from Microsoft, Citrix continues to lead the multiuser NT market with its MetaFrame and WinFrame products.

The MetaFrame software runs on a server and allows ICA thin clients to access Solaris-based applications. The applications execute entirely on the server, but the clients have access to everything as if they were running the application locally. MetaFrame only sends the display data across the network to the client, which allows clients with minimal processing power to execute applications they would normally not have access to.

Supported operating systems

Citrix uses its ICA protocol to deliver the applications. The architecture lets clients simultaneously run applications from both Windows NT and UNIX servers. Citrix has developed ICA clients for most desktop operating systems, which can make networking a much easier task for networks with many different operating systems. The clients have been developed for Windows, OS/2, Mac OS, Linux, and an assortment of UNIX systems (Solaris, AIX, HP-UX, OpenServer, and IRIX). In addition, it also supports a number of thin-client terminals like Wyse and Neoware, or it can give Web browser support to clients through Java or ActiveX. Due to the wide variety of platforms it can run on, MetaFrame has over 15 million end users and is supported in a variety of languages, making it the most popular application server in the world.

MetaFrame clustering

MetaFrame allows groups of Solaris servers to be joined together in clusters. The Citrix clustering technology runs at a superior level to the previously available options from Sun or Microsoft (for Solaris and Windows NT, respectively). Citrix has provided user access load balancing, but limits it to user access and doesn't support load balancing of the individual operating system. Server configuration is much the same, but MetaFrame does employ the Solaris Network Information System Plus (NIS+) to manage the same set of user accounts across multiple servers, much like MetaFrame uses domain controllers to keep a single set of user accounts under Windows NT.

MetaFrame has been scaled to over 120 servers in a single cluster under Windows NT, which will efficiently support thousands of simultaneous users. This number is obviously greater than the two-server maximum that NT supports natively, and it appears that there will be comparable levels of scalability for the Solaris option.

Multiprocessor support

Although Windows 2000 will increase the multiprocessing capabilities for the Windows platform, it's far behind Solaris, which can support up to 64 processors. Currently, Citrix is testing MetaFrame for Solaris in up to 32 processor Solaris servers. The advantage of multiprocessor support is cost savings while maintaining performance. Organizations will be able to save money by employing a number of smaller servers to equate the multi-user serving power of a more expensive, larger server.

Add-ons

A number of add-on products make MetaFrame particularly attractive. The first, and probably most interesting, is the Citrix Load-Balancing Services. It provides a way to balance incoming users across multiple servers in a single cluster. The next add-on that's worth mentioning is the SecureICA Services, which effectively and easily adds encryption between ICA clients and servers. Additionally, MetaFrame also has add-ons that analyze and tune the performance of clusters of servers, and to remotely install applications across a cluster of Citrix servers.

Currently, the add-on products are available only for the Windows NT version of MetaFrame, but will ultimately be available for Solaris. In fact, according to the Citrix Web site at www.citrix.com, they will release the Load-Balancing add-on with the first official release of the Solaris product.

Advantages of MetaFrame

Although competitive technologies exist, MetaFrame is superior in many ways. For instance, the current version of the X Windows Low-Bandwidth X (LBX) extension is inefficient for use on high-latency network connections such as a dialup connection. This is an area in which ICA has proven to perform very well. Another option that's currently available for Solaris is SunRay, which includes the HotDesk technology from Sun. MetaFrame has several advantages over the Sun alternative, but the most notable are the intuitive management tools.

Citrix hasn't released pricing information for the Solaris version of MetaFrame, which is projected to ship at press time. The price will probably be equivalent to the NT version. MetaFrame has considerable competition from companies like Sun and Microsoft. However, Citrix already has the trust of many system administrators running Windows NT, and the Solaris version of its software tremendously expands the possibilities of organizations that run both platforms on their networks. Because it simplifies the management of large numbers of users and servers, this package is certainly one to watch. ✳

Understanding run levels and /etc/rc2.d

by Edgar Danielyan

In most modern UNIX systems, including the Solaris Operating Environment, there's a concept of *run level* and various methods and tools to control a system's run level. In this article, we'll introduce all eight run levels available in Solaris and describe the second run level, which is the normal multiuser state of the system.

Run levels

A *run level* is a configuration set that defines the system's state—for example, which processes are to be run, which filesystems are to be mounted, etc. Run levels are set by /sbin/init, the process spawner that starts right after the process scheduler and always has PID 1. Init, in turn, is controlled by /etc/inittab, a text file that defines the run levels and specifies scripts to be executed for each level, in the following format:

```
ID:run_level:action_to_take:process_io
```

For example, the entry for the second run level looks like this:

```
s2:23:wait:/sbin/rc2 > /dev/msglog 2<>/dev/msglog
➥</dev/console
```

This line defines a run level called s2 that sets the system's state in second and third run levels by running /sbin/rc2 and redirecting output appropriately.

Table A: *Run levels*

Run level	Description
S I s	Single user state, usually used for system administration. Only local file systems are mounted, and access to the system is possible only via console. This run level doesn't require the existence of /etc/inittab.
0	This will shut down the system and run firmware (depending on the architecture).
1	Mostly like S run level, not frequently used.
2	The normal, everyday multiuser state of the system.
3	Like the second level, plus local resources are made available for access over network as configured.
4	The "alternative" multiuser state—usually not used.
5	Used to shut down the system and remove power, if possible.
6	Will halt the OS and reboot it to the state defined by initde fault in /etc/inittab.

Table B: *Typical files that are executed for the second run level*

/etc/rc2.d file name	Description
S01MOUNTFSYS	Mounts defined and configured file systems
S05RMTMPFILES	Removes temporary files from /tmp
S20sysetup	Provides nothing useful
S21perf	Runs system activity data collection daemon, if enabled
S30sysid.net	Configures basic networking
S47asppp	Starts asynchronous PPP daemon, if enabled
S69inet	Initializes TCP/IP
S70uucp	Initializes UUCP, if applicable
S71rpc	Starts Sun Remote Procedure Call (RPC) services
S71sysid.sys	Starts various system identification procedures
S72autoinstall	Starts Sun JumpStart routines; not needed on production systems
S72inetsvc	Starts Internet super-server and other daemons
S73cachefs.daemon	Starts cachefs file system daemon, if configured
S73nfs.client	Starts NFS client
S74autofs	Starts NFS automounter
S74syslog	Starts syslog daemon
S74xntpd	Starts Network Time Protocol (NTP) daemon, if configured
S75cron	Starts cron
S75savecore	Saves core file previously generated by a dump
S76nscd	Starts name services cache daemon
S80PRESERVE	Preserves files previously killed in vi
S80kdmconfig	Configures video drivers for X
S80lp	Starts printing service daemon (lp)
S80spc	Starts printing service (printd)
S85power	Initializes power management, if supported and configured
S88sendmail	Starts the sendmail daemon
S88utmpd	Starts utmpd
S92volmgt	Starts automated volume management
S93cacheos.finish	Completes postinstall; not necessary on a running system
S99audit	Starts auditing daemon, if enabled
S99dtlogin	Starts Common Desktop Environment login screen (dtlogin)

Description of available run levels

As we said earlier, there are eight run levels in the Solaris operating environment. See **Table A** for a description of each.

Look at the second run level

Let's take a look at the second run level, which is the usual multiuser state of the system. You can effect change to the second run level by either directly booting into the second run level, or by changing the run level (for example, from the single user (S) run level). The script that sets the second run level, /sbin/rc2, sequentially executes scripts located in the /etc/rc2.d directory, and when the last script completes, change to the second run level is considered complete. Script filenames in /etc/rc2.d and other rc directories follow this special naming format:

```
[S|K]NNalpha
```

where S and K, correspondingly, mean start and kill, NN is a number from 00 to 99, and alpha is a descriptive element (e.g., S69inet, S75cron, S92-volmgt). A logical sequence of execution is determined by the double-digit number after the first letter. All files that satisfy the above criteria are executed by /sbin/rc2 to reach the second run level. The usual files present in the /etc/rc2.d are (in order) are shown in **Table B** on the previous page.

That's it

You can add your own startup scripts to be executed if you follow the naming convention we described. We recommend using the sequence numbers 90-99, which means that your particular script will be executed after all required system daemons and processes are operating. Also, using a descriptive filename is a good idea—for example, S99apache and S98ntpdate are okay, but S99myscrpt isn't. ✳

Converting PDF files to HTML files with pdftohtml

by Guergui Ovtcharov

Nowadays, many CD-ROMs containing technical documentation use an HTML or PDF file format for the table of contents. Each entry contains a link to the appropriate PDF file or HTML. For example, there might be a link to the subdirectory PDFs/a007.pdf. When making the CD-ROM available on a Web server on the local network, the HTML table of contents works well.

When using a PDF table of contents, things get a little more complicated. The PDF file needs to be downloaded (e.g., in the local directory /tmp) and then the Web browser starts Acrobat Reader to view the downloaded file. But now, the links are local with respect to the downloaded file and not associated with the PDF files on the server any more. Clicking on links in the PDF table of contents fail now, because the example link now points to /tmp/PDFs/a007.pdf, instead of the subdirectory on the server. We wanted to preserve local links in PDF documents, so we set out to implement a PDF-to-HTML converter. In the process, however, we ultimately built a converter with even greater capabilities, including advanced formatting with layers for Netscape and IE.

What's pdftohtml?

Pdftohtml is an open source PDF-to-HTML converter that's based on the code in Derek Noonburg's Xpdf package. The latest version 0.22 is able to convert many features of the PDF document, like layout, links, images, and font faces (bold and italic). Both pdftohtml and Xpdf are distributed under the GNU Public License (GPL).

Installing pdftohtml

You can download the latest version of pdftohtml from **www.ra.informatik.uni-stuttgart.de/~gosho/pdftohtml/code.html**. For major releases, our open-source project usually provides binaries for Solaris and Linux. For minor releases, only source code is available (the current version is availabile in binary). If you want to compile the source code, you will need an ANSI C++ compiler. To compile a binary, it should be sufficient to compile the code using

```
$ make
```

Future releases will come with a configure script. After that, you should find the binary engine pdftohtml.bin and a wrapper script pdftohtml. You'll find that pdftohtml.bin handles the pure conversion, whereas the wrapper script handles the interaction with the operating environment. For example, in order to extract the images contained in the PDF file, you need pnmtopng in your `PATH` environment variable, or it could be hardcoded in the wrapper script.

Useful options

The default settings will generate a simple HTML document with plain text and links, and bold and italic font faces. To preserve the layout of the PDF document as closely as possible in the HTML document, use the -c option to generate a complex document. Other available options include:

- Option -i—Ignores images, no images are generated.
- Option -f—Doesn't generate frames if the option -c isn't given.
- Option -e <extension>—Set the extension for all image files generated from ppm and pbm images. The extension png is the default.

Customizing the image format

The binary pdftohtml.bin writes the standard image formats ppm, pbm, and JPEG. The wrapper script converts the file to something readable by a browser (png). Then, the names of all extracted ppm and pbm images are saved in the file images.log for the final conversion in the format. If the user needs another image format for the ppm and pbm images, he may replace line 87 in the wrapper script

```
pnmtopng $DIR/$ppmfile > $DIR/$file.png
```

with his converter (or series of converters). For convenience, the option -e changes the suffix from .png (default) to the desired suffix in the output HTML file.

Current limitations

In version 0.22, vector drawings in the PDF file are completely ignored; only pixmaps are extracted. This also affects tables, where only the content is extracted, but no separation lines. Also pdftohtml makes no attempt to reverse engineer the layout in the PDF file (e.g., no tables or lists are generated as HTML output).

When called with the -c option, pdftohtml uses layers supported by Netscape 4.0 and Internet Explorer 4.5 or higher. This means that older browsers may not be able to view a generated HTML document. Also, complex documents are somewhat fragile with respect to the font settings in the Web browser's preferences. We recommend using dynamic fonts. ✳

Further reading

For more information on the topics we discussed in this article, refer to the following Web sites:

- www.ra.informatik.uni-stuttgart.de/~gosho/pdftohtml
- www.ra.informatik.uni-stuttgart.de/~gosho/pdftohtml/code.html
- www.foolabs.com/xpdf/
- www.cdrom.com/pub/png/apps/pnmtopng.html
- www.gnu.org/copyleft/gpl.html

Configuring BIND 8

by Don Kuenz

Solaris uses BIND, the Berkeley Internet Name Domain system, to implement its Domain Name System (DNS) functionality. Sun started bundling BIND version 8 with Solaris 7. Earlier Solaris releases came bundled with BIND version 4. In this article, we'll show you how to configure BIND 8 and how to migrate from BIND 4.

DNS primarily maps host names to host IP addresses and vice versa. It maintains a database that contains matching pairs of host names and IP addresses. This allows you to administer host names by changing a centralized database located on one or more BIND servers. It saves you the effort of changing the /etc/hosts file on each and every host connected to your LAN.

BIND uses a client/server architecture. When a client needs to perform host name/IP address translation, it forwards either a host name or an IP address to a server. The server then searches its database or other servers for the requested pair before returning the missing half of the pair to the client.

Domain names

A fully qualified hostname contains a host name followed by a domain name. For instance, the host named **www.elementkjournals.com** uses www as an unqualified host name that belongs to a domain named elementkjournals.com. The host named **ftp.elementkjournals.com** also belongs to that same domain.

You can create your own unique domain name by registering it with an accredited Registrar. Use the search engine at **rs.internic.net/whois.html** to determine the uniqueness of your domain name. You can pick a registrar from **rs.internic.net/alpha.html**.

Listing A: *The partial contents of the two files necessary to enable a DNS client*

```
/etc/resolv.conf:
domain inside.biz
nameserver 192.42.172.130

/etc/nsswitch.conf:
hosts:    file dns
```

Listing B: *The output from an nslookup session*

```
# nslookup
Default Server:  apollo.inside.biz
Address:         192.42.172.130

> hyperion
Server:    apollo.inside.biz
Address:   192.42.172.130

Name:      hyperion.inside.biz
Address:   192.42.172.141

> hyperion.inside.biz
Server:    apollo.inside.biz
Address:   192.42.172.130

Name:      hyperion.inside.biz
Address:   192.42.172.141
> exit
#
```

Today, only a few top-level domains such as COM, NET, EDU, GOV, and ORG exist. In the near future you should see more top-level domains. For the purposes of this article, we'll use inside.biz as our domain name.

Configuring a DNS client

To enable DNS client functionality on a Solaris host, you must create a file named */etc/resolv.conf* and modify your /etc/nsswitch.conf file. Listing A shows the partial contents of both files.

The first file specifies inside.biz as our domain name. This causes Solaris to use inside.biz as our local domain. The first file also tells our DNS client to use the host assigned to 192.42.172.130 as a DNS server. The second file tells our host what to do when it needs to resolve a host name. As configured, Solaris first searches for the name in the /etc/hosts file. If that search fails, it then queries the name server specified in the /etc/resolv.conf file.

After you configure these two files, you can reboot your client host and start using DNS. Solaris includes a couple of tools named `nslookup` and `nstest` to help you debug your DNS installations. Listing B shows the output from an `nslookup` session. This session uses the greater than symbol (>) as a prompt. Our session shows three commands: `hyperion`, `hyperion.inside.biz`, and `exit`.

The first command causes `nslookup` to append our domain name, inside.biz, to the end host named apollo to form a fully qualified host name,

Listing C: *The output from an nstest session*

```
# nstest 192.42.172.130
> ?
a{host} - query  T_A
A{addr} - iquery T_A
b{user} - query  T_MB
B{user} - query  T_MG
f{host} - query  T_UINFO
g{host} - query  T_GID
G{gid}  - iquery T_GID
h{host} - query  T_HINFO
i{host} - query  T_MINFO
p{host} - query  T_PTR
m{host} - query  T_MX
M{host} - query  T_MAILB
n{host} - query  T_NS
r{host} - query  T_MR
s{host} - query  T_SOA
T{host} - query  T_TXT
u{host} - query  T_UID
U{uid}  - iquery T_UID
x{host} - query  T_AXFR
w{host} - query  T_WKS
F{host} - query  T_AFSDB
c{host} - query  T_CNAME
*{host} - query  T_ANY
> ahyperion.inside.biz
;; res_mkquery(0, hyperion.inside.biz, 1, 1)
;; res_send()
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 64729
;; flags: rd; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0
;;      hyperion.inside.biz, type = A, class = IN
;; Querying server (# 1) address = 192.42.172.130
;; got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 64729
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1
;;      hyperion.inside.biz, type = A, class = IN
hyperion.inside.biz.    1D IN A     192.42.172.141
inside.biz.             1D IN NS    apollo.inside.biz.
apollo.inside.biz.      1D IN A     192.42.172.130
>
#
```

apollo.inside.biz. It then queries our name server for the IP address that's paired with this name. The second command shows how we can use a fully qualified domain name to return the same IP address. Of course, the final command, `exit`, causes `nslookup` to terminate.

Solaris's other DNS tool, `nstest`, gives you a better picture of the client/server conversation that takes place during DNS lookups. Listing C shows the output from an `nstest` session. Notice that you enter the IP address of your DNS server as a command argument. Doing so shows three commands. The `nstest` tool also uses the greater than symbol (>) as a prompt. The first command, ?, causes `nstest` to display all of its commands. Actually, `nstest` displays all of the commands whenever it doesn't recognize the first character of a command. The second command, `ahyperion.inside.biz`, causes `nstest` to display address information about a host named hyperion.inside.biz. Unfortunately, `nstest` lacks an `exit` command, so you must type *control-d (^d)* to exit.

Configuring a DNS server

In order to create a DNS server, you must first enable its client functionality by configuring both the /etc/resolv.conf and the /etc/nsswitch files, as shown in the previous section. Next, you need to create a file named */etc/named.conf*. The BIND daemon, `named`, uses this file as input. Solaris automatically starts `named` when it boots, if it finds a valid /etc/named.conf file. Listing D shows the contents of our /etc/named.conf file.

BIND version numbers skip from version 4 to version 8. Versions 5 through 7 don't exist. The most visible differences between versions 4 and 8 are the name and format of the configuration file used by named. Version 8 uses /etc/named.conf, while version 4 uses a file named /etc/named.boot. You can migrate from version 4 to version 8 by using Solaris's `named-bootconf` command to convert /etc/named.boot to */etc/named.conf*.

The contents of /etc/named.conf provide the key to understanding how to configure a DNS server. We'll spend the remainder of this article discussing this file line by line.

Listing D: *Contents of the /etc/named.conf file*

```
options {
        ///etc/named.conf
        //
        //boot file for primary name
        // server
        //
        //type domain source file or host
        //
        directory        "/var/named";

};

zone "inside.biz" in {
        type master;
        file "db.inside";
};

   zone "172.42.192.in-addr.arpa" in {
        type master;
        file "db.192.42.172";
};

   zone "0.0.127.in-addr.arpa" in {
        type master;
        file "db.127.0.0";
};

   zone "." in {
        type hint;
        file "named.ca";
};
```

Listing E: *Contents of the /var/named/db.inside file*

```
; db.inside - hostname to IP address
; resolution table
@ IN SOA apollo.inside.biz.
 root.apollo.inside.biz. (
 97011001 ; Serial number
 10800 ; Refresh after three hours
 3600 ; Retry after one hour
 604800 ; Expire after one week
 86400 ) ; Minimum TTL of one day
 IN NS apollo.inside.biz.

; Define the localhost
localhost IN A 127.0.0.1
; Define the hosts in this zone
apollo IN A 192.42.172.130
aphrodite IN A 192.42.172.132
zeus IN A 192.42.172.136
hyperion IN A 192.42.172.141
eos IN A 192.42.172.145
hermes IN A 192.42.172.146

; Add CNAME records below, as desired
; (for host aliases)
;loghost IN CNAME apollo.inside.biz.

; Add MX records (mail exchanger) records
; below.
inside.biz. IN MX 0 apollo.inside.biz.
```

The first group of lines in /etc/named.conf instruct `named` to look in the /var/named directory for configuration files. The remaining lines configure a domain named inside.biz that uses IP addresses in the 192.42.172.xx range. For your own DNS server, you need to replace inside.biz with your domain name and replace the 192.42.172.xx IP addresses with your own IP addresses.

Hostname to IP address translation

DNS servers need to return a matching IP address whenever a client queries them with a hostname. The next group of lines in /etc/resolv.conf, starting with `zone "inside.biz"`, tell `named` that it can find hostname to IP address maps for our inside.biz domain in the /var/named/db.inside file. Listing E on the previous page, shows the contents of this file.

This file shows that we use a DNS server named apollo.inside.biz. It also shows that five other hosts (aphrodite, zeus, hyperion, eos, and hermes) exist in the same domain with apollo. All of these hosts co-exist on a single ethernet segment and use IP addresses that begin with 192.42.172.

To tailor this file to your needs, you need to replace inside.biz with your own domain name. You also need to substitute your own host names and IP addresses for the ones shown in Listing E.

IP address to hostname translation

DNS servers need to return a matching hostname whenever a client queries them with an IP address. This is known as *reverse-mapping*. The next group of lines in /etc/resolv.conf, starting with `zone "172.42.192.in-addr.arp"` and continuing with `zone "0.0.127.in-addr.arpa"`, instruct `named` to use `/var/named/db.192.42.172` and `/var/named/db.127.0.0` as reverse-mappings to obtain a hostname from an IP address. Listing F shows the contents of the /var/named/db.192.42.172 file and Listing G shows the contents of the /var/named/db.127.0.0 file.

Please note the rather curious way that we specify IP addresses in both files. If you look carefully, you'll see that we enter IP addresses in reverse order. We enter 192.42.172.130 as 130.172.42.192, and so on. You must follow this convention in your own reverse-mapping files. Again, to use these files on your DNS server, you need to replace inside.biz with your own domain name, substitute your hostnames for the hostnames shown, and replace the IP addresses with your own IP addresses.

Using root name servers

Whenever DNS servers fail to locate hostname and IP addresses within their own, local database, they need to query outside DNS servers. The last group of lines in /etc/resolv.conf, starting with `zone "."`, tell `named` that it can find root name servers in the /var/named/named.ca file. Listing H shows the contents of this file.

DNS servers use root servers to discover mappings that fall outside of their own domain. You can use this file as is on your own DNS server. As mentioned in the file, you can obtain updated copies of this file from ftp.rs.internic.net.

Conclusion

Solaris uses BIND 8 to implement DNS functionality. In this article, we discussed domain names and BIND's architecture. We've also shown you how to configure BIND 8 to use it as a DNS client and a DNS server. ✳

Listing F: *Contents of the /var/named/db.192.42.172 file*

```
@ IN SOA  apollo.inside.biz. root.apollo.inside.biz. (
   97011001          ; Serial number
   10800             ; Refresh after 3 hrs
   3600              ; Retry after one hour
   604800            ; Expire after one week
   86400 )           ; Minimum TTL of one day
   IN NS   apollo.inside.biz.

130.172.42.192.in-addr.arpa.    IN PTR   apollo.inside.biz.
132.172.42.192.in-addr.arpa.    IN PTR   aphrodite.inside.biz.
136.172.42.192.in-addr.arpa.    IN PTR   hyperion.inside.biz.
141.172.42.192.in-addr.arpa.    IN PTR   zeus.inside.biz.
145.172.42.192.in-addr.arpa.    IN PTR   eos.inside.biz.
146.172.42.192.in-addr.arpa.    IN PTR   hermes.inside.biz.
;145                            IN PTR   eos.inside.biz.
```

Listing G: *Contents of the /var/named/db.127.0.0 file*

```
; db.127.0.0
@ IN SOA  apollo.inside.biz.
 root.apollo.inside.biz. (
 97011002          ; Serial number
 10800             ; Refresh after 3 hrs
 3600              ; Retry after one hour
 604800            ; Expire after one week
 86400 )           ; Minimum TTL of one day
 IN NS   apollo.inside.biz.
1                IN PTR  localhost.inside.biz.
```

Listing H: *Contents of the /var/named/named.ca file*

```
options {
;       This file holds the information on root name servers need
;       initialize cache of Internet domain name servers
;       (e.g., reference this file in the "cache  .  <file>"
;       configuration file of BIND domain name servers).
;
;       This file is made available by InterNIC registration serv
;       under anonymous FTP as
;           file                /domain/named.root
;           on server           FTP.RS.INTERNIC.NET
;       -OR- under Gopher at    RS.INTERNIC.NET
;           under menu          InterNIC Registration Services (N
;              submenu          InterNIC Registration Archives
;           file                named.root
;
;       last update:    Aug 22, 1997
;       related version of root zone:   1997082200
;
;
; formerly NS.INTERNIC.NET
;
.                        3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.      3600000      A     198.41.0.4
;
; formerly NS1.ISI.EDU
;
.                        3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.      3600000      A     128.9.0.107
;
; formerly C.PSI.NET
;
.                        3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.      3600000      A     192.33.4.12
;
; formerly TERP.UMD.EDU
;
.                        3600000      NS    D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.      3600000      A     128.8.10.90
;
; formerly NS.NASA.GOV
;
.                        3600000      NS    E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET.      3600000      A     192.203.230.10
;
; formerly NS.ISC.ORG
;
.                        3600000      NS    F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.      3600000      A     192.5.5.241
;
; formerly NS.NIC.DDN.MIL
;
.                        3600000      NS    G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET.      3600000      A     192.112.36.4
;
; formerly AOS.ARL.ARMY.MIL
;
.                        3600000      NS    H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.      3600000      A     128.63.2.53
;
; formerly NIC.NORDU.NET
;
.                        3600000      NS    I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.      3600000      A     192.36.148.17
;
; temporarily housed at NSI (InterNIC)
;
.                        3600000      NS    J.ROOT-SERVERS.NET.
J.ROOT-SERVERS.NET.      3600000      A     198.41.0.10
;
; housed in LINX, operated by RIPE NCC
;
.                        3600000      NS    K.ROOT-SERVERS.NET.
K.ROOT-SERVERS.NET.      3600000      A     193.0.14.129
;
; temporarily housed at ISI (IANA)
;
.                        3600000      NS    L.ROOT-SERVERS.NET.
L.ROOT-SERVERS.NET.      3600000      A     198.32.64.12
;
; housed in Japan, operated by WIDE
;
.                        3600000      NS    M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.      3600000      A     202.12.27.33
; End of File
        ///etc/named.conf
```

Easily create multiple levels of directories

A great thing about the Solaris `mkdir` command is that you can easily create multiple levels of directories with one command.

For instance, let's assume that you have a directory named /home/al/cad. You just gained a new customer named acme, and decide that you'd like to create your new files in an organized manner. Assuming that /home/al/cad is the only directory that exists, you can type

```
mkdir -p /home/al/cad/customer/acme/project1
```

to create the customer, acme, and project1 directories with one easy command.

Create a one-drive partition

by Jerry L.M. Phillips

Have you ever wanted to format a disk drive and create just one partition that represents the entire available space on the drive? Carefully follow the simple technique we describe in this article, and you can dispense with multiple partitions (slices) on your disk drive.

One drive, one partition

Let's say that you connect a new Sun UltraSCSI 4.2 GB disk drive to your Sun machine. After installing the drive, you may need to reboot the machine by shutting it down and issuing the `boot -r` command at the `Ok>` prompt. Next, you execute the `format` command as follows and determine which disk drive it is that you wish to reconfigure:

```
# format
Searching for disks╓done
AVAILABLE DISK SELECTIONS:
0. c0t0d0 <ST39140A cyl 17660 alt 2 hd 16 sec 63>
➡/pci@1f,0/pci@1,1/ide@3/dad@0,0
1. c0t1d0 <ST39140A cyl 17660 alt 2 hd 16 sec 63>
➡/pci@1f,0/pci@1,1/ide@3/dad@1,0
2. c1t5d0 <SUN9.0G cyl 4924 alt 2 hd 27 sec 133>
➡/pci@1f,0/pci@1/scsi@4,1/sd@5,0
3. c2t3d0 <SUN4.2G cyl 3880 alt 2 hd 16 sec 135>
➡/pci@1f,0/pci@1/scsi@2/sd@3,0
Specify disk (enter its number): 3
Selecting c2t3d0
[disk formatted]
```

Listing A: *The output of our verify command*

```
Primary label contents:
Volume name = <        >
ascii name  = <SUN4.2G cyl 3880 alt 2 hd 16 sec 135>
pcyl       = 3882
ncyl       = 3880
acyl       = 2
nhead      = 16
nsect      = 135
Part  Tag   Flag  CylindersSize             Blocks
  0root     wm 0 - 121              128.67MB (122/0/0) 263520
  1swap     wu 122 - 243   128.67MB (122/0/0) 263520
  2backup   wu 0 ╨ 3879             4.00GB   (3880/0/0) 8380800
  3unassigned  wm 0                 0        (0/0/0) 0
  4unassigned  wm 0                 0        (0/0/0) 0
  5unassigned  wm 0                 0        (0/0/0) 0
  6usr   wm 244 - 3879     3.74GB            (3636/0/0) 7853760
  7unassigned  wm 0                 0        (0/0/0) 0

format> partition
```

Entering the number 3 selects disk drive 3 and displays the following format menu:

```
FORMAT MENU:
disk         - select a disk
type         - select (define) a disk type
partition    - select (define) a partition table
current      - describe the current disk
format       - format and analyze the disk
repair       - repair a defective sector
label        - write label to the disk
analyze      - surface analysis
defect       - defect list management
backup       - search for backup labels
verify       - read and display labels
save         - save new disk/partition/definitions
inquiry      - show vendor, product, and revision
volname      - set 8-character volume name
!<cmd>       - execute <cmd>, then return
quit

format> verify
```

Entering the `verify` command in the format menu lists the characteristics of the disk drive that you select. **Listing A** shows us the current layout of our drive.

Entering the partition command displays the partition menu. You use commands in this menu to modify, label, and save your new partition. In this case, you want to create one 4.00-GB partition representing the entire available space on the disk drive:

```
PARTITION MENU:
0        - change `0' partition
1        - change `1' partition
2        - change `2' partition
3        - change `3' partition
4        - change `4' partition
5        - change `5' partition
6        - change `6' partition
7        - change `7' partition
select   - select a predefined table
modify   - modify a predefined partition table
name     - name the current table
print    - display the current table
label    - write partition map and label to disk
!<cmd>   - execute <cmd>, then return
quit

partition> modify
```

Entering the `modify` command in the partition menu allows you to alter the existing format characteristics. **Listing B** on the next page shows the steps needed to make your drive one partition.

Your disk drive, c2t3d0s0, now contains one partition that's 4.00 GB in size. The next step is to create a file system on that partition using the `newfs` command:

```
# newfs /dev/rdsk/c2t3d0s0
newfs: construct a new file system =>/dev/rdsk/c2t3d0s0:(y/n)? y
```

After you type *y* to create the file system, the program lists the disk drive characteristics and identifies the super block and super block spares as it creates them. If you want to optimize the disk drive for either time or space, you need to experiment with the `tunefs` command. For example, if you want to change the current optimization setting from time to space, you can type the command:

```
# tunefs -o space /dev/dsk/c2t3d0s0
```

About our contributors

Clayton E. Crooks II is a self-employed computer consultant living in Knoxville, TN. He's married with one child. His hobbies include game development, 3-D modeling and any athletic activity he can find time for.

Edgar Danielyan is currently self-employed. His qualifications include Cisco Certified Network Associate, Diploma in Company Law from the British Institute of Legal Executives, and certified paralegal from the University of Southern Colorado. He has been working as a network administrator and manager of a top-level domain of Armenia. He's also worked for the United Nations, the ministry of defense, a national telco, a bank, and has been a partner in a law firm. He speaks four languages, likes good tea, and is a member of the ACM, IEEE CS, USENIX, CIPS, ISOC, and IPG clubs, to name a few. He can be reached at **edd@danielyan.com**.

Don Kuenz works at Computing Resources Company (**gtcs.com/crc**). They provide programming, administration, and hardware for Sun and PC platforms. You can reach Don at **kuenz@gtcs.com**.

Gueorgui Ovtcharov has been studying computer science at the University of Stuttgart since 1997. During this time he did C++ programming on Solaris and Solaris administration in the Computer Architecture Lab of the University of Stuttgart.

Jerry L.M. Phillips, M.S. is director of the Database Center at Eastern Virginia Medical School. In addition to his administrative duties, he manages Sun/Solaris based platforms for the medical school, including DNS, sendmail, WWW, anonymous ftp, proxy, and library servers. Jerry is also an Oracle DBA and manages Oracle production and development database servers for business and student information system services running on Sun/Solaris servers as well.

Paul A. Watters is the project manager at Neuroflex, where he's responsible for developing natural language database systems in Java on the Solaris platform. He can be reached at **pwatters@mpce.mq.edu.au**.

Inside Solaris

Tips & Techniques for users of Sun Solaris

Inside Solaris (ISSN 1081-3314) is published monthly by Element K Journals, a division of Element K Press, 500 Canal View Boulevard, Rochester, NY 14624.

Customer Relations

US toll free	(800) 223-8720
Outside of the US	(716) 240-7301
Customer Relations fax	(716) 214-2386

For subscriptions, fulfillment questions, and requests for group subscriptions, address your letters to

Element K Journals Customer Relations
500 Canal View Boulevard
Rochester, NY 14623

Or contact Customer Relations via Internet email at journals@element-k.com.

Editorial

Editor	Garrett Suhm
Assistant Editor	Jill Suhm
Managing Editor	Michelle Rogers
Copy Editors	Rachel Krayer Glenna Lechner
Contributing Editors	Clayton E. Crooks II Edgar Danielyan Don Kuenz Gueorgui Ovtcharov Jerry L.M. Phillips, M.S. Paul A. Watters
Print Designer	Rachel J. King
Cover and Content Design	Melissa Ribaudo

You may address tips, special requests, and other correspondence to

The Editor, *Inside Solaris*
500 Canal View Boulevard
Rochester, NY 14623

Editorial Department fax	(716) 272-0064

Or contact us via Internet email at inside_solaris@elementkjournals.com.

Sorry, but due to the volume of mail we receive, we can't always promise a reply, although we do read every letter.

Element K Journals

General Manager	Kelly Baptiste
Manager of Customer Relations	Nicole Pate
Manager of Operations	Cristal Haygood
Manager of Print Design	Ian Caspersson
Manager of Product Marketing	Mike Mayfield
Senior Product Marketing Manager	Brian Cardona

Postmaster

Periodicals postage paid in Rochester, NY and additional mailing offices.

Postmaster: Send address changes to

Inside Solaris
P.O. Box 92880
Rochester, NY 14692

Price

Domestic	$129/yr ($11.00 each)
Outside US	$149/yr ($13.00 each)

Our Canadian GST# is: R140496720. CPM# is: 1446703. QST# is: 1018491237.

Back Issues

To order a back issue from the last six months, call Customer Relations at (800) 223-8720. Back issues cost $11.00 each, $13.00 outside the US. You can pay with MasterCard, VISA, Discover, or American Express.

Are you moving?

If you've moved recently or you're planning to move, you can guarantee uninterrupted service on your subscription by calling us at (800) 223-8720 and giving us your new address. Or you can fax us your label with the appropriate changes at (716) 214-2386. Our Customer Relations department is also available via email at journals@element-k.com.

Coming up...

- Proxy caching with Squid
- Sniffing your network

PERIODICALS MAIL

2096

```
*****************************************************MIXED ADC 400
C:7661905    00002096    04/01                                   1
RUDOLPH LIEDTKE                                                  1
RJL SYSTEMS
33955 HARPER AVE
CLINTON TOWNSHIP, MI  48035-4218
```

USPS ARMIN PSI 881 APPROVED POLY

Listing B: *Steps for making your drive one partition*

```
Select partitioning base:
        0. Current partition table (original)
        1. All Free Hog
Choose base (enter number) [0]? 1

Part      Tag    Flag    Cylinders    Size          Blocks
 0       root     wm     0 - 121     128.67MB   (122/0/0)  263520
 1       swap     wu     122 - 243   128.67MB   (122/0/0)  263520
 2     backup     wu     0 - 3879    4.00GB     (3880/0/0) 8380800
 3  unassigned    wm     0            0         (0/0/0) 0
 4  unassigned    wm     0            0         (0/0/0) 0
 5  unassigned    wm     0            0         (0/0/0) 0
 6        usr     wm     244 - 3879 3.74GB      (3636/0/0) 7853760
 7  unassigned    wm     0            0         (0/0/0) 0

Do you wish to continue creating a new partition table based on
above table[yes]? yes
Free Hog partition[6]? 0
Enter size of partition '1' [0b, 0c, 0.00mb, 0.00gb]: 0
Enter size of partition '3' [0b, 0c, 0.00mb, 0.00gb]: 0
Enter size of partition '4' [0b, 0c, 0.00mb, 0.00gb]: 0
Enter size of partition '5' [0b, 0c, 0.00mb, 0.00gb]: 0
Enter size of partition '6' [0b, 0c, 0.00mb, 0.00gb]: 0
Enter size of partition '7' [0b, 0c, 0.00mb, 0.00gb]: 0
Part      Tag    Flag    Cylinders       Size          Blocks
 0       root     wm     0 - 3879        4.00GB     (3880/0/0) 8380800
 1       swap     wu     0               0          (0/0/0) 0
 2     backup     wu     0 - 3879        4.00GB     (3880/0/0) 8380800
 3 unassigned     wm     0               0          (0/0/0) 0
 4 unassigned     wm     0               0          (0/0/0) 0
 5 unassigned     wm     0               0          (0/0/0) 0
 6        usr     wm     0               0          (0/0/0) 0
 7 unassigned     wm     0               0          (0/0/0) 0

Okay to make this the current partition table[yes]?
Enter table name (remember quotes): "4.00gb"
Ready to label disk, continue? yes
Partition> quit
Format> quit
```

You can also change the minimum amount of reserved space held back from users. The default value you typically encounter on the newer disk drives is 1 percent. The following command changes the minimum amount to 5 percent reserved space:

```
# tunefs -m 5 /dev/dsk/c2t3d0s0
```

It's time to create a mount point for the new disk drive:

```
# mkdir /test
```

Finally, you can mount the drive:

```
# mount /dev/dsk/c2t3d0s0 /test
```

You're all set to go. Don't forget to edit /etc/vfstab to mount the new disk drive automatically the next time that you reboot the system.

Conclusion

In this article, we've shown you a trick with the format command that allows you to dispense with multiple partitions on your new disk drive. You can create a single partition on your disk drive that represents all of the available space on the drive. This can simplify the disk structure when you don't need partitions—for example, when loading news articles, building Oracle data files, and so on. Knowing tricks like this will set you apart as a Solaris system administrator. ✳